the pyramid is a lie

FrOSCon 2015

@rtens_
This talk is about

Automated testing

Software Architecture
Full Disclosure

heavily inspired by

"Modelling by example"
by everzet
Me: Nikolas Martens

Engineer  Coach  Consultant
I ❤ Automated Testing

"Of course. How else could you program?"
Integration

"Integration Tests are a Scam"
- jbrains

Unit

"TDD is dead. Long live testing."
- DHH
Integration testing

- through browser
- running webserver
  - including databases
  - + external services
Integration testing

```php
$this->visit('/products');
$product = $this->find('css', 
    ".products li:contains('Red Car')");
$product->click('Add to basket');
$this->assertNotNull($this->find('css', 
    ".basket li:contains('Red Car')"));
```
Unit testing

- Single class/method
- Most/all collaborators faked
  - "One reason to fail"
Unit testing

$product = \texttt{new} \text{ Product('Red Car');}
$basket = \texttt{new} \text{ Basket();}
$basket->add($product);
$this->assertContains($product, $basket->getProducts());
Integration -vs- Unit
confidence
Succeeding with Agile - Mike Cohn

Top down → Unit

<table>
<thead>
<tr>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
</tr>
<tr>
<td>UI</td>
</tr>
</tbody>
</table>

Bottom up
Feature A
Feature B
Feature C

Test

Triple "Coverage"
What about
Domain layer testing

$productCatalogue->addProduct('Red Car', 7);
$productCatalogue->addProduct('Blue Car', 5);
$basketService->addProduct('Red Car');
$basketService->addProduct('Blue Car');
$this->assertEquals(12, $basketService->currentSum());
Domain layer testing

Given the Product "Red Car" priced 7 Euro
Given the Product "Blue Car" priced 5 Euro

When I add "Red Car" and "Blue Car" to the basket

Then the sum of the basket should be 12 Euro

ubiquitous language
Ubiquitous language

When I add "Red Car" to the basket

Service context

```php
$basketService->addProduct('Red Car');
```

UI context

```php
$product = $this->find('css', ".products li:contains('Red Car')");
$product->click('Add to basket');
```
or
both
How to?

Cucumber: worlds
Behat: suites
PHPUnit: data providers
class SomeTest extends PHPUnit_Framework_TestCase {

    /** @dataProvider driveServiceAndUi */
    function testStuff(SomeTestDriver $driver) {
        // ...
        $driver->whenIAdd_ToTheBasket('Red Car');
        // ...
    }

    function driveServiceAndUi() {
        return [
            [new SomeTestServiceDriver()],
            [new SomeTestUiDriver()]
        ];
    }
}
interface SomeTestDriver {

    public function whenIAdd_ToTheBasket($product);
}

class SomeTestServiceDriver implements SomeTestDriver {

    public function whenIAdd_ToTheBasket($product) {
        $this->basketService->addProduct($product);
    }
}

class SomeTestUiDriver implements SomeTestDriver {

    public function whenIAdd_ToTheBasket($product) {
        $product = $this->crawler->find("#$product");
        $product->click('Add to basket');
    }
}
Recap
1. Write tests on domain level
2. Use ubiquitous language
3. Implement on multiple levels
4. Profit!
Further Information

"Architecture - The lost Years"
Uncle Bob

"Modelling by Example"
everzet

rtens.org